FM-

68-FM-260

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

MSC INTERNAL NOTE NO. 68-FM-260

November 29, 1968

A FINITE DIFFERENCING METHOD OF COMPUTING THE STATE TRANSITION MATRIX FOR ANY TYPE OF TRAJECTORY MODEL, CONIC THROUGH N-BODY

Advanced Mission Design Branch

MISSION PLANNING AND ANALYSIS DIVISION

MANNED SPACECRAFT CENTER
HOUSTON, TEXAS

(NASA-TM-X-69671) A FINITE DIFFERENCING METHOD OF COMPUTING THE STATE TRANSITION MATRIX FOR ANY TYPE OF TRAJECTORY MODEL, CONIC THROUGH (NASA) 18 p

N74-70525

Unclas
00/99 16327

MSC INTERNAL NOTE NO. 68-FM-260

A FINITE DIFFERENCING METHOD OF COMPUTING THE STATE TRANSITION MATRIX FOR ANY TYPE OF TRAJECTORY MODEL, CONIC THROUGH N-BODY

By Ellis W. Henry
Advanced Mission Design Branch

November 29, 1968

MISSION PLANNING AND ANALYSIS DIVISION
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
MANNED SPACECRAFT CENTER
HOUSTON, TEXAS

Approved: Jack Funk
Jack Funk, Chief
Advanced Mission Design Branch

Approved:
John P. Mayer, Chief
Mission Planning and Analysis Division

CONTENTS

A FINITE DIFFERENCING METHOD OF COMPUTING THE STATE TRANSITION MATRIX FOR ANY TYPE OF TRAJECTORY MODEL, CONIC THROUGH N-BODY

By Ellis W. Henry

SUMMARY

This paper presents a method of computing a state transition matrix by finite differencing differentiation. The method uses a state vector propagation routine explicitly and is shown to be independent of trajectory modeling simplification or complexity. Thus it is applicable to an extensive range of problems. A method of numerical qualification is discussed to show how to achieve maximum accuracy as well as to determine that accuracy realistically. The predominant emphasis throughout is toward applications on a digital computer rather than rigorous mathematical considerations.

Other methods of computing the state transition matrix are mentioned to show the relative advantages and limitations. Application of the method to problems in addition to or instead of transition matrix computation are suggested.

INTRODUCTION

The state transition matrix is becoming increasingly applicable in various type of trajectory computer programs including navigation, guidance, orbit determination, and error analysis. Briefly, the state transition matrix relates mathematically the effects of propagating errors (or uncertainties) from one instant of time to another. That is, for a specific trajectory model and an initial state vector and time interval, the state transition matrix can be computed and shows a linear approximation to the change from a nominal trajectory resulting from arbitrary errors or uncertainties.

It is possible for certain simple trajectory models to compute this matrix by analytic formulations. Even in the simplest model (two-body conics), the analytic expressions can be formidable as well as restricted in application (ref. 1). Only slightly more complex models can be solved analytically, and then generally only by some approximation or simplification. These may still have restricted application.

The method described in this paper involves partial derivitives computed by the method of finite differences and is justified through a combination of intuition, mathematical reason, and computer application. It will be seen to be applicable to conic trajectories, matched conics, and oblate planets, and can accommodate N-body perturbations, impulsive or finite thrusting, etc. The technique in itself is not new (ref. 2). This paper is intended first to describe the technique, then to develop a method of determining its accuracy and/or region of applicability, and finally to show that it can be made completely independent of the complexity of the trajectory model. The end product is a concise computer program with nearly unlimited application.

This method is not intended to be competitive with analytic formulation as far as computation speed when analytic formulations are adequate. However, this relatively simple formulation provides an ideal check (test cases) on much more complicated analytic versions, and, as stated, is readily applicable to otherwise impossibly difficult models (i.e., state of the art numerical integration trajectory programs).

The appendix gives a sample version of such a program, namely, a FORTRAN subroutine, which uses a trajectory subprogram explicity, and thus is applicable to any trajectory model independent of the model complexity.

DESCRIPTION OF THE STATE TRANSITION MATRIX AND DEFINITION OF TERMINOLOGY

The state transition matrix, hereafter called the PHI matrix, or Φ, is a 6-by-6 matrix of partial derivitives of a state vector at the time t_f with respect to the state vector at time t_o.

By definition, if the initial or nominal state at time t_o is

$$\bar{\bar{S}}_o = (\bar{r}_o, \bar{v}_o) = X_o\hat{i} + Y_o\hat{j} + Z_o\hat{k},\ \dot{X}_o\hat{i} + \dot{Y}_o\hat{j} + \dot{Z}_o\hat{k} \tag{1}$$

($\hat{i}$, $\hat{j}$, and $\hat{k}$ are unit vectors) and the nominal state at time t_f is defined by

$$\bar{\bar{S}}_f = (\bar{r}_f, \bar{v}_f) = X_f\hat{i} + Y_f\hat{j} + Z_f\hat{k},\ \dot{X}_f\hat{i} + \dot{Y}_f\hat{j} + \dot{Z}_f\hat{k} \tag{2}$$

then the PHI matrix is

$$\Phi\ (t_f, t_o) = \begin{bmatrix}
\frac{\partial X_f}{\partial X_o} & \frac{\partial X_f}{\partial Y_o} & \frac{\partial X_f}{\partial Z_o} & \frac{\partial X_f}{\partial \dot{X}_o} & \frac{\partial X_f}{\partial \dot{Y}_o} & \frac{\partial X_f}{\partial \dot{Z}_o} \\
\frac{\partial Y_f}{\partial X_o} & \frac{\partial Y_f}{\partial Y_o} & \frac{\partial Y_f}{\partial Z_o} & \frac{\partial Y_f}{\partial \dot{X}_o} & \frac{\partial Y_f}{\partial \dot{Y}_o} & \frac{\partial Y_f}{\partial \dot{Z}_o} \\
\frac{\partial Z_f}{\partial X_o} & \frac{\partial Z_f}{\partial Y_o} & \frac{\partial Z_f}{\partial Z_o} & \frac{\partial Z_f}{\partial \dot{X}_o} & \frac{\partial Z_f}{\partial \dot{Y}_o} & \frac{\partial Z_f}{\partial \dot{Z}_o} \\
\frac{\partial \dot{X}_f}{\partial X_o} & \frac{\partial \dot{X}_f}{\partial Y_o} & \frac{\partial \dot{X}_f}{\partial Z_o} & \frac{\partial \dot{X}_f}{\partial \dot{X}_o} & \frac{\partial \dot{X}_f}{\partial \dot{Y}_o} & \frac{\partial \dot{X}_f}{\partial \dot{Z}_o} \\
\frac{\partial \dot{Y}_f}{\partial X_o} & \frac{\partial \dot{Y}_f}{\partial Y_o} & \frac{\partial \dot{Y}_f}{\partial Z_o} & \frac{\partial \dot{Y}_f}{\partial \dot{X}_o} & \frac{\partial \dot{Y}_f}{\partial \dot{Y}_o} & \frac{\partial \dot{Y}_f}{\partial \dot{Z}_o} \\
\frac{\partial \dot{Z}_f}{\partial X_o} & \frac{\partial \dot{Z}_f}{\partial Y_o} & \frac{\partial \dot{Z}_f}{\partial Z_o} & \frac{\partial \dot{Z}_f}{\partial \dot{X}_o} & \frac{\partial \dot{Z}_f}{\partial \dot{Y}_o} & \frac{\partial \dot{Z}_f}{\partial \dot{Z}_o}
\end{bmatrix}_{t_f,\ t_o} \tag{3}$$

Note for emphasis that the first column is the partial derivitive of the state vector at time t_f with respect to parameter X_o. Similarly the second column is with respect to Y_o , etc.

Frequently the literature shows the PHI matrix as "partitioned", or made up of four 3-by-3 matrices; i.e.

$$\Phi = \left[\begin{array}{c|c} \dfrac{\partial \bar{r}_f}{\partial \bar{r}_o} & \dfrac{\partial \bar{r}_f}{\partial \bar{v}_o} \\ \hline \dfrac{\partial \bar{v}_f}{\partial \bar{r}_o} & \dfrac{\partial \bar{v}_f}{\partial \bar{v}_o} \end{array}\right] \qquad (4)$$

If we assume for the moment that the PHI matrix can be obtained and if we assign errors or uncertainties to each component of $\bar{\bar{S}}_o$, which I shall call δ_o,

$$\bar{\bar{\delta}}_o = \delta\bar{r}_o, \ \delta\bar{v}_o \qquad (5)$$

Then the propagative effects of the initial δ_o over the time interval are

$$\bar{\bar{\delta}}_f = (\bar{\delta}r_f, \ \bar{\delta}v_f) = \Phi \ \bar{\bar{\delta}}_o \qquad (6)$$

BASIS OF NUMERICAL COMPUTATION OF THE PHI MATRIX

Consider any element of equation (3); for example, $\partial X_f/\partial X_o$. From the definition of a derivative,

$$\frac{\partial X_f}{\partial X_o} = \lim_{\Delta X \to 0} \frac{(X_f + \Delta X_f) - X_f}{(X_o + \Delta X_o) - X_o} = \lim_{\Delta X \to 0} \frac{(X_f + \Delta X_f) - X_f}{\Delta X_o} \qquad (7)$$

If X_f can be evaluated from a functional relationship involving X_o, then $(X_f + \Delta X_f)$ is obtained from $(X_o + \Delta X_o)$. To obtain the partial derivitive [equation (7)] numerically, we can choose a relatively small value of ΔX_o and using the functional relationship evaluate the right-hand side of equation (7). If ΔX_o could be made arbitrarily smaller, then the numerical derivitive approaches the precise mathematical value. On a digital computer, the value of ΔX_o cannot approach zero too closely due to truncation to a finite number of digits. It will be shown, however, that a sufficiently small value can be used which yields results sufficiently accurate for most requirements. All 36 elements of the PHI matrix can be expressed similarly to equation (7).

ACTUAL COMPUTATION OF THE PHI MATRIX - ASSUMPTIONS AND PROCEDURES

At this point, it is necessary and convenient to assume that a state vector propagation routine is available, so that we can write

$$\overline{\overline{S}}_f = G(\overline{\overline{S}}_o, \tau) \tag{8}$$

where τ represents the time interval, Δt, and perhaps an actual date, if required by the model.

For convenience, equation (8) may be written with the notation $(X_f, Y_f, Z_f, \dot{X}_f, \dot{Y}_f, \dot{Z}_f,) = G\ (X_o, Y_o, Z_o, \dot{X}_o, \dot{Y}_o, \dot{Z}_o, \tau)$. These notations will be used interchangably. The equations simply imply that, given a state vector $\overline{\overline{S}}_o$ (by which we mean a state vector at any instant of time), we can find the corresponding state vector at any other instant of time ($\tau = \Delta t$). G implies some trajectory model of an unspecified degree of complexity which functionally relates $\overline{\overline{S}}_f$ to $\overline{\overline{S}}_o$ through a time interval. Specifically for computer application, we assume G is a subroutine which accepts as inputs the initial state vector and time interval, along with whatever other inputs are required to specify uniqueness, and produces as an output the resulting $\overline{\overline{S}}_f$. The FORTRAN notation might be CALL G (SO, DT, SF).

We are now equiped with the basic formulation and techniques to compute Φ [equation (3)] by the method of equation (7).

We first note that for equation (7), we need to compute X_f from X_o using our subroutine G, i.e., $(X_f, Y_f, Z_f, \dot{X}_f, \dot{Y}_f, \dot{Z}_f) = G\ (X_o, Y_o, Z_o, \dot{X}_o, \dot{Y}_o, \dot{Z}_o, \tau)$. We have also obtained $Y_f, Z_f \ldots \dot{Z}_f$ which will be required for the second, third, . . ., sixth column evaluations.

When we replace X_o with $(X_o + \Delta X_o)$ in our subroutine G, we obtain $(X_f + \Delta X_f, Y_f + \Delta Y_f, Z_f + \Delta Z_f, \dot{X}_f + \Delta\dot{X}_f, \dot{Y}_f + \Delta\dot{Y}_f, \dot{Z}_f + \Delta\dot{Z}_f) = G\ (X_o + \Delta X_o, Y_o, Z_o, \dot{X}_o, \dot{Y}_o, \dot{Z}_o, \tau)$, where ΔX_f is interpreted as the change in X_f due to a small change in X_o with Y_o . . . $\dot{Z}_o$ unchanged. Without elaboration, it should be noted that we now have not only those values to evaluate equation (7), but also all quantities in column one of equation (3) by similar expressions. Now if we evaluate $G\ (X_o,\ Y_o + \Delta Y_o,\ Z_o,\ \dot{X}_o,\ \dot{Y}_o,\ \dot{Z}_o,\ \tau)$ we can compute all elements in column two, and so on through column six.

To review the steps taken in the actual evaluation of the PHI matrix, we used the G subroutine once to obtain the nominal state vector at the time t_f. All of these six output values were used; each was used in only one column, but in every element (row) in that column.

Then the G routine was used six more times. Each use involved changing only one of the nominal inputs by a small amount, but this change produced changes in all six of the output values, thus allowing the computation of all the elements in an appropriate column.

The subroutine G was applied seven times. (In general, if an N × N matrix is required, a subroutine would be called upon N + 1 times.) By efficient FORTRAN coding, using DO loops and subscript notation, the above wordy description takes on a very neat and concise mathematical appearance. An example appears in the appendix - excluding but assuming a subroutine G.

APPLICATION AND SELF-CHECK TEST CASES

Equation (6) shows one of the simplest applications of the PHI matrix It can be used to test, qualify, and calibrate the PHI matrix computation.

Presumably, if we compute the PHI matrix and assume a set of deviations $\overline{\overline{\delta}}_o$ (errors or uncertainties) in the nominal trajectory, then those deviations propagate by equation (6) over the interval Δt to produce

$$\overline{\overline{\delta}}_f = \Phi\ \overline{\overline{\delta}}_o \tag{9}$$

Effectively we predict the deviations at time t_f based upon an assumed set existing at time t_o, using Φ. But given that assumed set, we can also use the G subroutine [equation (8)] to determine the actual deviations directly; i.e.,

$$(\overline{\overline{S}}_f + \overline{\overline{\delta}}_f) \doteq G\ \left[(\overline{\overline{S}}_o + \overline{\overline{\delta}}_o),\ \tau\right] \tag{10}$$

Thus,

$$\overline{\overline{\delta}}_f = (\overline{\overline{S}}_f + \overline{\overline{\delta}}_f) - \overline{\overline{S}}_f \tag{11}$$

The values of δ_f produced with equations (10) and (11) are mathematically precise; they do not depend on any linearity assumptions and properly include any cross-coupling effects. They are numerically as accurate as our trajectory computation model (SUBROUTINE G).

The comparable values produced with equation (9) depend upon the applicability of our linear theory PHI matrix, (i.e., δ_o small enough to apply linear theory) as well as upon the accuracy in computing the PHI matrix itself.

The qualification of the PHI matrix hinges upon the agreement between the actual and predicted δ_f for the same δ_o. This self-check test will be used to evaluate the accuracy of the method on a computer.

This procedure is preferred to some of the identity tests applicable to a unique formulation for two reasons:

1. The described process is universally applicable and shows realistically the numerical accuracy of the computed quantities.

2. Even though an identity test may be applicable (such as a determinate of PHI $\equiv 0$), precise equivalence cannot be obtained on a finite digits computer, and that resulting discrepancy may not be relatable to errors in the computation of the matrix or the test.

NUMERIC QUALIFICATION - DETERMINATION OF THE INCREMENT VALUES

Referring back to equation (7) and the associated text, it is apparent that the accuracy of the PHI matrix dependents upon choosing $\Delta X, \ldots, \Delta\dot{Z}$ sufficiently small to assume a linear region and yet large enough to insure significance in the computation. Thus a tradeoff is required. The comparison of the previous paragraph is used to empirically determine the best compromise.

To remain independent of the units of the initial state vector, the value of the Δ's in equation (7) were obtained as a percentage of the $\bar{r}_o$, $\bar{v}_o$ magnitudes; i.e.,

$$\left.\begin{aligned} \Delta X = \Delta Y = \Delta Z = PR\ |\bar{r}_o| \\ \Delta\dot{X} = \Delta\dot{Y} = \Delta\dot{Z} = PR\ |\bar{v}_o| \end{aligned}\right\} \qquad (12)$$

where PR (perturbation ratio) is varied parametrically for best agreement between actual and predicted δ_f's when the δ_o's were of a size comparable to the Δ's.

To fix at least an approximate value of the Δ's for use on a digital computer, let us begin by considering the accuracy of the G subroutine. For discussion, let's assume the output of the G subroutine is, say, 14 digits of significance (perhaps not unreasonable for a double precision routine). The numerator of equation (7) is obtained from the differences in two solutions produced with the G routine. The denominator is the amount that the inputs were changed to obtain the different outputs. If the input Δ's are so small that only the least significant digit of each output is changed, then the difference in two solutions has only one digit of accuracy; thus PHI is correct to one significant digit. We surely require more accuracy than this, so the Δ's must be increased. The PHI matrix elements can be no more accurate than the significant digits in the difference of two solutions. The input ΔV's must be large enough to change several digits of the output.

At the other extreme we could make the Δ's so large that all 14 of the significant digits change. However, this surely stretches our linearity assumptions. We may try for half as many digits in the PHI matrix as significant in the G subroutine, then empirically test for adequacy or improvement in results. Since $(S_f + \Delta_f) = G(S_o + \Delta_o)$, a reasonable assumption is then $|\overline{\overline{\Delta}}_f| \simeq |\overline{\overline{\Delta}}_o|$ within one or two orders of magnitude. We may hope to obtain seven digit accuracy in $\overline{\overline{\Delta}}_f$. Thus we begin by assuming $PR = 1 \times 10^{-7}$ in equation (12) then vary this value parameterically (by powers of 10) for best agreement between equations (9) and (11). The next section of this paper shows the results of this empirical calibration.

ACTUAL NUMERIC ACCURACY IN PROTOTYPE PROGRAM

While it is impossible to generalize "accuracy" without limiting the class of problem to be solved and size of the Δt interval, a fairly reasonable expectation for general application in the discussed program is to use $PR = 5 \times 10^{-10}$. The predicted and actual δ's agree to six or seven digits, depending on the Δt interval; thus, the PHI matrix has this order of accuracy.

The problems considered to obtain these empirical values included circular orbits, high energy ellipses, and hyperbolas of the lunar and interplanetary scale. They included Δt intervals of a fraction of a day up to several hundred days. In the case of elliptic trajectories, the Δt intervals were frequently many revolutions, and for hyperbolas crossed spheres of influence. Further, the G subroutine models included conic models (Kepler problem) and numerically integrated solutions (double precision, Cowell method) to conic equations of motion, oblate planet potential functions, and N-body perturbations.

The accuracy suggested above results from a compromise to obtain generality.

For applications which limit the general class to a more specific class, say, for earth-centered hyperbolic trajectories with time intervals less than three days, then the accuracy can be improved by two or three significant digits.

It is easy and worthwhile to determine PR empirically for whatever class problem and G subroutine is appropriate. For this reason, the PHI matrix routine shown in the appendix allows PR as an input, and optionally includes sufficient printout to determine a quantitative choice of various PR's for best results and calibration. Obviously, PR could be varied within the subroutine and a suitable test included to select the best value used. This is not done to simplify and to keep computation time at a minimum.

CONCLUDING REMARKS

The chief advantage of the numerical method presented is simplicity and applicability to any trajectory program; the limitations may be computational speed or limited accuracy. In the areas of study where state transition matrices are applied, there are computationally faster analytical programs for simpler problems and more accurate programs for complex problems. At the simple extreme, a conic trajectory program, an analytic formulation (ref. 1) may be more accurate and more efficient when applicable. As the conic model is improved by including select perturbation forces (ref. 3), analytic formulation becomes more difficult and, eventually, prohibitive.

At the complex extreme are precision integrated trajectories which include N-bodies and oblate planets' spheres of influence, solar pressure, thrust, drag, etc. These may require integration of the PHI matrix along with the equations of motion in order to retain sufficient accuracy and indeed may be more efficient as well.

In between these two extremes are the hundreds of intermediate trajectory programs, better than conic but not precise. The numerical method presented is best suited for these programs, although applicable at both extremes. It is independent of the complexity of the model, the formulation, and the features included or omitted. For example, navigation, guidance, impulse thrust, or finite thrust with steering logic, etc., may be a part of the G routine.

This method was developed to generate test cases for analytic formulations of the simplest form, and to extend local capabilities to more complex problems. It is certain to find some application at either extreme, but most frequently in the intermediate range.

This paper has dealt exclusively with the state transition matrix in Cartesian coordinate form, namely the 6-by-6 matrix discussed in current literature. For whatever other coordinate systems or parameters are meaningful to derive input and output, a variational matrix (not necessarily of square dimensions) can be computed using the numerical method discussed here.

Two examples are

1. A six-degree-of-freedom problem necessitating a 12-by-12 matrix if attitude and state vector are correlated through navigation, thrusting, and guidance - an extension of the present PHIMAT routine given a G subroutine.

2. A problem of relating a certain set of orbital (osculating) element deviations at one instant of time to a similar or different set of deviations at another.

APPENDIX

FORTRAN SUBROUTINE PHIMAT

APPENDIX

FORTRAN SUBROUTINE PHIMAT

```
SUBROUTINE PHIMAT (SO, DT, PR, PHI, ETC.)
DOUBLE PRECISION SO, DT, PR, PHI, TINC, SOP, SFP, ERR
DIMENSION SO (6), PHI (6, 6), TINC (6), SIP (6), SFP (6), ERR (6)
IF (PR.EQ.0) PR = 5.0D-10
TINC (1) - PR *DSQRT (SO (1) **2 + SO (2) **2 + SO (3) **2)
TINC (4) = PR *DSQRT (SO (4) **2 + SO (5) **2 + SO (6) **2)
TINC (2) = TINC (1)
TINC (3) = TINC (1)
TINC (5) = TINC (4)
TINC (6) = TINC (4)
C  FOLLOWING CALL IS SYMBOLIC.  ACTUAL CALL DEPENDS ON ROUTINE USED.
   CALL G (SO, DT, SF)
   DO 20 I = 1, 6
   DO 10 J = 1, 6
C  SOP, SFP INDICATE PERTURBED SO, SF
10 SOP (J) = SO(J) (J)
   SOP (I) = SOP (I) TINC (I)
   CALL G (SOP, DT, SFP)
   DO 20 K = 1, 6
20 PHI (K, I) = (SFP(K) - SF (K)) / TINC (I)
C  RETURN AT THIS POINT UNLESS CALIBRATION TESTING REQUIRED
C  ASSUME INITIAL ERRORS IN SO ARE TINC
```

```
      PRINT 200, TINC
C     COMPUTE ERRORS AFTER DT USING PHI, ASSUMING A MATRIX TIMES
C     VECTOR ROUTING VIA NEXT CALL
      CALL MXV (PHI, TINC, ERR)
      PRINT 200, ERR
C     COMPUTE ERROR AFTER DT DIRECTLY USING SUBROUTINE G
      DO 50 I = 1, 6
50    SOP (I) = SO (I) + TINC (I)
      CALL G (SOP, DT, SFP)
      DO 60 I = 1, 6
60    ERR (I) = SFP (I) - SF (I)
      PRINT 200, ERR
C     ACTUAL AND PREDICTED ERRORS HAVE BEEN PRINTED
200   FORMAT (6D22.11)
      RETURN
      END
```

REFERENCES

1. Bond, Victor R.; and Faust, Nicholas L.: An Analytical Formulation of the Conic State Transition Matrix Using a Universal Conic Variable. MSC IN 66-FM-99, September 19, 1966.

2. Savely, Robert T.; and Phillips, Laurel A.: A Study of Free-Flight Analytic State Partials for Error Propagation. MSC IN 65-FM-88, June 25, 1965.

3. Kirkpatrick, James C.: An Analytic Method for Computing the State Transition Matrix Which Includes the First-Order Secular Perturbations of Planet Oblateness. MSC IN 68-FM-261, October 18, 1968.